Readlink

Value written in the output buffer is not null-terminated and may not contain the entire file name

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-02

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 3903 bytes

<table>
<tr><td><b>Attack Category</b></td><td><ul><li>Path spoofing or confusion problem</li></ul></td></tr>
<tr><td><b>Vulnerability Category</b></td><td><ul><li>No Null Termination</li><li>Indeterminate File/Path</li></ul></td></tr>
<tr><td><b>Software Context</b></td><td><ul><li>Filename Management</li></ul></td></tr>
<tr><td><b>Location</b></td><td></td></tr>
<tr><td><b>Description</b></td><td>The readlink() function attempts to get the filename of the file pointed to by the given link.
The value written in the output buffer is not null-terminated.
Also, if the return value is the same as the size of the input buffer, then it is possible that the buffer does not contain the entire file name.</td></tr>
<tr><td><b>APIs</b></td><td><table><tr><th>Function Name</th><th>Comments</th></tr><tr><td>readlink</td><td></td></tr></table></td></tr>
<tr><td><b>Method of Attack</b></td><td>The issue here is that the value written into the output buffer may not be the entire filename.
The filename is truncated if the length of the output buffer is smaller than the length of the filename.
Also, the filename is not null-terminated. This is not a vulnerability in itself, but subsequent use of functions such as strcpy() may cause buffer overflows (due to a lack of null-termination). Also, unexpected results could occur if the filename is truncated, and the programmer does not expect it.</td></tr>
<tr><td><b>Exception Criteria</b></td><td></td></tr>
<tr><td><b>Solutions</b></td><td><table><tr><th>Solution Applicability</th><th>Solution Description</th><th>Solution Efficacy</th></tr><tr><td>Generally applicable to any readlink.</td><td>Always check the return value of readlink().
If the return value is equal to the length of the buffer, then</td><td>Effective.</td></tr></table></td></tr>
</table>

[1] http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

<table>
<tr><td></td><td>
<table>
<tr><td></td><td>make the buffer larger and retry.</td><td></td></tr>
<tr><td>Generally applicable to any readlink.</td><td>Always null-terminate the value returned by readlink.</td><td>Effective.</td></tr>
</table>
</td></tr>
<tr><td><b>Signature Details</b></td><td>int readlink(const char *filename, char *buffer, size_t size);</td></tr>
<tr><td><b>Examples of Incorrect Code</b></td><td>
<pre>
...
char buffer[100];
readlink (filename, buffer, 100);
printf("The file name is: %s\n",
buffer);
...
</pre>
</td></tr>
<tr><td><b>Examples of Corrected Code</b></td><td>
<pre>
char *readlink_malloc (const char
*filename)
{
int size = 100;

while (1) {
char *buffer = (char *) malloc
(size);
int nchars = readlink (filename,
buffer, size);
if (nchars < 0)
return NULL;
if (nchars < size) {
buffer[nchars] = '\0';
return buffer;
}
free (buffer);
size *= 2;
}
}
</pre>
</td></tr>
<tr><td><b>Source References</b></td><td>• ITS4 Source Code Vulnerability Scanning Tool [2]
• http://www.gnu.org/software/libc/manual/html_node/Symbolic-Links.html</td></tr>
<tr><td><b>Recommended Resource</b></td><td></td></tr>
<tr><td><b>Discriminant Set</b></td><td>
<table>
<tr><td><b>Operating System</b></td><td>• UNIX (All)</td></tr>
<tr><td><b>Languages</b></td><td>• C
• C++</td></tr>
</table>
</td></tr>
</table>

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

[1] mailto:copyright@cigital.com